Application of Information Theory, Lecture 4

Asymptotic Equipartition Property, Data Compression & Gambling

Iftach Haitner

Tel Aviv University.

November 18, 2014

Part I

Asymptotic Equipartition Theorem

Entropy as # of bits to describe random variable

Entropy as # of bits to describe random variable

- ▶ In what sense is it true?

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$\binom{n}{k} \quad := \quad \frac{n!}{k!(n-k)!)}$$

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \le n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$\binom{n}{k} \quad := \quad \frac{n!}{k!(n-k)!)}$$

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \quad \text{(Stirling approx: } m! \approx (\frac{m}{e})^m)
\end{aligned}
$$

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \le n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \quad \text{(Stirling approx: } m! \approx (\frac{m}{e})^m) \\
&= \frac{n^n}{k^k(n-k)^{n-k}}
\end{aligned}
$$

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \quad \text{(Stirling approx: } m! \approx (\frac{m}{e})^m\text{)} \\
&= \frac{n^n}{k^k (n-k)^{n-k}} \\
&= \left(\frac{k}{n}\right)^{-k} \cdot \left(\frac{n-k}{n}\right)^{-(n-k)}
\end{aligned}
$$

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \quad \text{(Stirling approx: } m! \approx (\frac{m}{e})^m) \\
&= \frac{n^n}{k^k(n-k)^{n-k}} \\
&= \left(\frac{k}{n}\right)^{-k} \cdot \left(\frac{n-k}{n}\right)^{-(n-k)} \\
&= p^{-pn} \cdot (1-p)^{-(1-p)n}
\end{aligned}
$$

Entropy as # of bits to describe random variable

▶ In what sense is it true?

▶ Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \quad \text{(Stirling approx: } m! \approx (\frac{m}{e})^m) \\
&= \frac{n^n}{k^k(n-k)^{n-k}} \\
&= \left(\frac{k}{n}\right)^{-k} \cdot \left(\frac{n-k}{n}\right)^{-(n-k)} \\
&= p^{-pn} \cdot (1-p)^{-(1-p)n} \\
&= 2^{-p\log(p)n} \cdot 2^{-(1-p)\log(1-p)n}
\end{aligned}
$$

Entropy as # of bits to describe random variable

- ▶ In what sense is it true?
- ▶ Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \quad \text{(Stirling approx: } m! \approx (\frac{m}{e})^m) \\
&= \frac{n^n}{k^k(n-k)^{n-k}} \\
&= \left(\frac{k}{n}\right)^{-k} \cdot \left(\frac{n-k}{n}\right)^{-(n-k)} \\
&= p^{-pn} \cdot (1-p)^{-(1-p)n} \\
&= 2^{-p\log(p)n} \cdot 2^{-(1-p)\log(1-p)n} \\
&= 2^{n(-p\log p - (1-p)\log(1-p))}
\end{aligned}
$$

Entropy as # of bits to describe random variable

- ▶ In what sense is it true?
- ▶ Let $k \le n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \quad \text{(Stirling approx: } m! \approx (\frac{m}{e})^m) \\
&= \frac{n^n}{k^k(n-k)^{n-k}} \\
&= \left(\frac{k}{n}\right)^{-k} \cdot \left(\frac{n-k}{n}\right)^{-(n-k)} \\
&= p^{-pn} \cdot (1-p)^{-(1-p)n} \\
&= 2^{-p\log(p)n} \cdot 2^{-(1-p)\log(1-p)n} \\
&= 2^{n(-p\log p - (1-p)\log(1-p))} \\
&= 2^{n \cdot h(p)}
\end{aligned}
$$

Entropy as # of bits to describe random variable

- In what sense is it true?
- Let $k \leq n \in \mathbb{N}$ and $p = \frac{k}{n}$

$$
\begin{aligned}
\binom{n}{k} &:= \frac{n!}{k!(n-k)!)} \\
&\approx \frac{(\frac{n}{e})^n}{(\frac{k}{e})^k \cdot (\frac{n-k}{e})^{n-k}} \text{ (Stirling approx: } m! \approx (\frac{m}{e})^m) \\
&= \frac{n^n}{k^k(n-k)^{n-k}} \\
&= \left(\frac{k}{n}\right)^{-k} \cdot \left(\frac{n-k}{n}\right)^{-(n-k)} \\
&= p^{-pn} \cdot (1-p)^{-(1-p)n} \\
&= 2^{-p\log(p)n} \cdot 2^{-(1-p)\log(1-p)n} \\
&= 2^{n(-p\log p - (1-p)\log(1-p))} \\
&= 2^{n \cdot h(p)}
\end{aligned}
$$

- It takes about $n \cdot h(p)$ bits to describe a string of k zeros in $\{0,1\}^n$.

Entropy as # of bits to describe random variable, cont.

Entropy as # of bits to describe random variable, cont.

▶ Let $x_1, \ldots, x_n$ be iid $\sim (p, 1 - p)$

Entropy as # of bits to describe random variable, cont.

- ▶ Let $x_1, \ldots, x_n$ be iid $\sim (p, 1 - p)$
- ▶ w.h.p. about pn of x_i's are zeros (law of large numbers)

Entropy as # of bits to describe random variable, cont.

- ► Let $x_1, \ldots, x_n$ be iid $\sim (p, 1 - p)$
- ► w.h.p. about pn of x_i's are zeros (law of large numbers)
- ► Assume that exactly $k = pn$ of x_i's are zeros

Entropy as # of bits to describe random variable, cont.

- Let $x_1, \ldots, x_n$ be iid $\sim (p, 1-p)$
- w.h.p. about pn of x_i's are zeros (law of large numbers)
- Assume that exactly $k = pn$ of x_i's are zeros
- There are $\binom{n}{k \approx 2^{nh(p)}}$ possibilities.

Entropy as # of bits to describe random variable, cont.

- Let $x_1, \ldots, x_n$ be iid $\sim (p, 1-p)$
- w.h.p. about pn of x_i's are zeros (law of large numbers)
- Assume that exactly $k = pn$ of x_i's are zeros
- There are $\binom{n}{k \approx 2^{nh(p)}}$ possibilities.
- We need $nh(p)$ to tell in which possibility we are.

Entropy as # of bits to describe random variable, cont.

- ▶ Let $x_1, \ldots, x_n$ be iid $\sim (p, 1 - p)$
- ▶ w.h.p. about pn of x_i's are zeros (law of large numbers)
- ▶ Assume that exactly $k = pn$ of x_i's are zeros
- ▶ There are $\binom{n}{k \approx 2^{nh(p)}}$ possibilities.
- ▶ We need $nh(p)$ to tell in which possibility we are.
- ▶ In other words: it takes about $nh(p)$ bits to describe $X = x_1, \ldots, x_n$, which is $H(X)$!

Entropy as # of bits to describe random variable, cont.

- ▶ Let $x_1, \ldots, x_n$ be iid $\sim (p, 1-p)$
- ▶ w.h.p. about pn of x_i's are zeros (law of large numbers)
- ▶ Assume that exactly $k = pn$ of x_i's are zeros
- ▶ There are $\binom{n}{k \approx 2^{nh(p)}}$ possibilities.
- ▶ We need $nh(p)$ to tell in which possibility we are.
- ▶ In other words: it takes about $nh(p)$ bits to describe $X = x_1, \ldots, x_n$, which is $H(X)$!
- ▶ Describing X:

Entropy as # of bits to describe random variable, cont.

- Let $x_1, \ldots, x_n$ be iid $\sim (p, 1 - p)$
- w.h.p. about pn of x_i's are zeros (law of large numbers)
- Assume that exactly $k = pn$ of x_i's are zeros
- There are $\binom{n}{k \approx 2^{nh(p)}}$ possibilities.
- We need $nh(p)$ to tell in which possibility we are.
- In other words: it takes about $nh(p)$ bits to describe $X = x_1, \ldots, x_n$, which is $H(X)$!
- Describing X:
  - Send k — the number of zeros in X. ($\log n$ bits)

Entropy as # of bits to describe random variable, cont.

- Let $x_1, \ldots, x_n$ be iid $\sim (p, 1 - p)$

- w.h.p. about pn of x_i's are zeros (law of large numbers)

- Assume that exactly $k = pn$ of x_i's are zeros

- There are $\binom{n}{k \approx 2^{nh(p)}}$ possibilities.

- We need $nh(p)$ to tell in which possibility we are.

- In other words: it takes about $nh(p)$ bits to describe $X = x_1, \ldots, x_n$, which is $H(X)$!

- Describing X:
  - Send k — the number of zeros in X.   (log n bits)
  - Send the index of X in the strings of k zeroes.   (about $H(X)$ bits)

Entropy as # of bits to describe random variable, cont.

- Let $x_1, \ldots, x_n$ be iid $\sim (p, 1-p)$
- w.h.p. about pn of x_i's are zeros (law of large numbers)
- Assume that exactly $k = pn$ of x_i's are zeros
- There are $\binom{n}{k \approx 2^{nh(p)}}$ possibilities.
- We need $nh(p)$ to tell in which possibility we are.
- In other words: it takes about $nh(p)$ bits to describe $X = x_1, \ldots, x_n$, which is $H(X)$!
- Describing X:
    - Send k — the number of zeros in X. (log n bits)
    - Send the index of X in the strings of k zeroes. (about $H(X)$ bits)
- Over all it takes about $H(X)$ bits

Entropy as # of bits to describe random variable, cont..

Entropy as # of bits to describe random variable, cont..

- Let $k_1, \ldots, k_\ell$ with $\sum k_i = n$, and let $p_i = \frac{k_i}{n}$

Entropy as # of bits to describe random variable, cont..

- Let $k_1, \ldots, k_\ell$ with $\sum k_i = n$, and let $p_i = \frac{k_i}{n}$
- $\binom{n}{k_1, \ldots, k_\ell} \approx 2^{n \cdot H(p_1, \ldots, p_\ell)}$

Entropy as # of bits to describe random variable, cont..

- ▶ Let $k_1, \ldots, k_\ell$ with $\sum k_i = n$, and let $p_i = \frac{k_i}{n}$
- ▶ $\binom{n}{k_1, \ldots, k_\ell} \approx 2^{n \cdot H(p_1, \ldots, p_\ell)}$
- ▶ Let $x_1, \ldots, x_n$ be iid $\sim (p_1, \ldots, p_\ell)$, and $n >> \ell$

Entropy as # of bits to describe random variable, cont..

- ► Let $k_1, \ldots, k_\ell$ with $\sum k_i = n$, and let $p_i = \frac{k_i}{n}$
- ► $\binom{n}{k_1, \ldots, k_\ell} \approx 2^{n \cdot H(p_1, \ldots, p_\ell)}$
- ► Let $x_1, \ldots, x_n$ be iid $\sim (p_1, \ldots, p_\ell)$, and $n >> \ell$
- ► w.h.p. we can describe $X = x_1, \ldots, x_n$ using $H(X) = n \cdot H(p_1, \ldots, p_\ell)$ bits.

Entropy as # of bits to describe random variable, cont..

- ▶ Let $k_1, \ldots, k_\ell$ with $\sum k_i = n$, and let $p_i = \frac{k_i}{n}$

- ▶ $\binom{n}{k_1, \ldots, k_\ell} \approx 2^{n \cdot H(p_1, \ldots, p_\ell)}$

- ▶ Let $x_1, \ldots, x_n$ be iid $\sim (p_1, \ldots, p_\ell)$, and $n >> \ell$

- ▶ w.h.p. we can describe $X = x_1, \ldots, x_n$ using $H(X) = n \cdot H(p_1, \ldots, p_\ell)$ bits.

  - ▶ $\forall j \in [\ell]$: Send the number of x_i's that get the value j.   ($\ell \cdot \log n$ bits)

Entropy as # of bits to describe random variable, cont..

- Let $k_1, \ldots, k_\ell$ with $\sum k_i = n$, and let $p_i = \frac{k_i}{n}$

- $\binom{n}{k_1, \ldots, k_\ell} \approx 2^{n \cdot H(p_1, \ldots, p_\ell)}$

- Let $x_1, \ldots, x_n$ be iid $\sim (p_1, \ldots, p_\ell)$, and $n >> \ell$

- w.h.p. we can describe $X = x_1, \ldots, x_n$ using $H(X) = n \cdot H(p_1, \ldots, p_\ell)$ bits.

  - $\forall j \in [\ell]$: Send the number of x_i's that get the value j.　($\ell \cdot \log n$ bits)
  - Send the index of X among all strings of this characterization. (about $H(X)$ bits)

Entropy as # of bits to describe random variable, cont..

- Let $k_1, \ldots, k_\ell$ with $\sum k_i = n$, and let $p_i = \frac{k_i}{n}$

- $\binom{n}{k_1, \ldots, k_\ell} \approx 2^{n \cdot H(p_1, \ldots, p_\ell)}$

- Let $x_1, \ldots, x_n$ be iid $\sim (p_1, \ldots, p_\ell)$, and $n >> \ell$

- w.h.p. we can describe $X = x_1, \ldots, x_n$ using $H(X) = n \cdot H(p_1, \ldots, p_\ell)$ bits.

    - $\forall j \in [\ell]$: Send the number of x_i's that get the value j.   ($\ell \cdot \log n$ bits)
    - Send the index of X among all strings of this characterization. (about $H(X)$ bits)

- Over all it takes about $H(X)$ bits

Asymptotic equipartition theorem (AEP)

Asymptotic equipartition theorem (AEP)

▶ A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n\to\infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n \to \infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = E\, X_1$.

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in **probability** to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n \to \infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathrm{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n\to\infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathrm{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n \to \infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathrm{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

- Example $X_1 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$ and $X_2 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in ⟨probability⟩ to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n\to\infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathrm{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

- Example $X_1 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$ and $X_2 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$

- $(X_1, X_2) = \left\{ \begin{array}{ll} 00, & .01 \\ 01, & .09 \\ 10, & .09 \\ 11, & .81 \end{array} \right.$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n \to \infty} \Pr\left[|Z_n - \mu| > \varepsilon\right] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathbb{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

- Example $X_1 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$ and $X_2 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$

- $(X_1, X_2) = \left\{ \begin{array}{ll} 00, & .01 \\ 01, & .09 \\ 10, & .09 \\ 11, & .81 \end{array} \right.$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n\to\infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathsf{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

- Example $X_1 = \begin{cases} 0, & .1 \\ 1, & .9 \end{cases}$ and $X_2 = \begin{cases} 0, & .1 \\ 1, & .9 \end{cases}$

- $(X_1, X_2) = \begin{cases} 00, & .01 \\ 01, & .09 \\ 10, & .09 \\ 11, & .81 \end{cases}$ and $p(X_1, X_2) = \begin{cases} .01, & .01 \\ .09, & .18 \\ .81, & .81 \end{cases}$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n \to \infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathbb{E} X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

- Example $X_1 = \begin{cases} 0, & .1 \\ 1, & .9 \end{cases}$ and $X_2 = \begin{cases} 0, & .1 \\ 1, & .9 \end{cases}$

- $(X_1, X_2) = \begin{cases} 00, & .01 \\ 01, & .09 \\ 10, & .09 \\ 11, & .81 \end{cases}$ and $p(X_1, X_2) = \begin{cases} .01, & .01 \\ .09, & .18 \\ .81, & .81 \end{cases}$

- $\log p(X_1, \ldots, X_n) = \sum_i \log p(X_i)$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n \to \infty} \Pr[|Z_n - \mu| > \varepsilon] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathsf{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

- Example $X_1 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$ and $X_2 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$

- $(X_1, X_2) = \left\{ \begin{array}{ll} 00, & .01 \\ 01, & .09 \\ 10, & .09 \\ 11, & .81 \end{array} \right.$ and $p(X_1, X_2) = \left\{ \begin{array}{ll} .01, & .01 \\ .09, & .18 \\ .81, & .81 \end{array} \right.$

- $\log p(X_1, \ldots, X_n) = \sum_i \log p(X_i)$

- Hence,
  $\mathsf{E}_{X_1, \ldots, X_n} [-\log p(X_1, \ldots, X_n)] = -\sum_i \mathsf{E}[\log p(X_i)] = H(X_1, \ldots, X_n)$

Asymptotic equipartition theorem (AEP)

- A sequence $\{Z_i\}_{i=1}^{\infty}$ of rv's converges in probability to μ (denoted $Z_n \xrightarrow{P} \mu$), if $\lim_{n \to \infty} \Pr\left[|Z_n - \mu| > \varepsilon\right] = 0$ for all $\varepsilon > 0$

- Let $X_1, \ldots, X_n$ be iid $\sim p$ and let $\mu = \mathrm{E}\, X_1$.

- *Weak law of large numbers*: $\frac{1}{n} \cdot \sum_{i=1}^{n} X_i \xrightarrow{P} \mu$

- Let $p(x_1, \ldots, x_n) = \prod_i p(x_i)$ and consider the rv $p(X_1, \ldots, X_n)$.

- Example $X_1 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$ and $X_2 = \left\{ \begin{array}{ll} 0, & .1 \\ 1, & .9 \end{array} \right.$

- $(X_1, X_2) = \left\{ \begin{array}{ll} 00, & .01 \\ 01, & .09 \\ 10, & .09 \\ 11, & .81 \end{array} \right.$ and $p(X_1, X_2) = \left\{ \begin{array}{ll} .01, & .01 \\ .09, & .18 \\ .81, & .81 \end{array} \right.$

- $\log p(X_1, \ldots, X_n) = \sum_i \log p(X_i)$

- Hence,
  $\mathrm{E}_{X_1, \ldots, X_n}\left[-\log p(X_1, \ldots, X_n)\right] = -\sum_i \mathrm{E}\left[\log p(X_i)\right] = H(X_1, \ldots, X_n)$

- We will show that w.h.p. $-\log p(X_1, \ldots, X_n)$ is close to its expectation

Asymptotic equipartition theorem (AEP), cont.

Asymptotic equipartition theorem (AEP), cont.

▶ By weak law of large numbers:

$\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathsf{E} \log p(X_1)$

Asymptotic equipartition theorem (AEP), cont.

▶ By weak law of large numbers:

$\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathsf{E} \log p(X_1)$

Asymptotic equipartition theorem (AEP), cont.

▶ By weak law of large numbers:

$\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathrm{E} \log p(X_1) = -H(X_1)$

Asymptotic equipartition theorem (AEP), cont.

- By weak law of large numbers:

  $\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathsf{E} \log p(X_1) = -H(X_1)$

- That is, $\lim_{n \to \infty} \Pr\left[\left|-\frac{1}{n} \log(p(X_1, \ldots, X_n)) - H(X_1)\right| > \varepsilon\right] = 0$, for any $\varepsilon > 0$

Asymptotic equipartition theorem (AEP), cont.

▶ By weak law of large numbers:

$\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathsf{E} \log p(X_1) = -H(X_1)$

▶ That is, $\lim_{n \to \infty} \Pr\left[\left|-\frac{1}{n} \log(p(X_1, \ldots, X_n)) - H(X_1)\right| > \varepsilon\right] = 0$, for any $\varepsilon > 0$

▶ Hence, $\forall \varepsilon > 0$

Asymptotic equipartition theorem (AEP), cont.

- By weak law of large numbers:

  $\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathbb{E} \log p(X_1) = -H(X_1)$

- That is, $\lim_{n \to \infty} \Pr \left[\left| -\frac{1}{n} \log(p(X_1, \ldots, X_n)) - H(X_1) \right| > \varepsilon \right] = 0$, for any $\varepsilon > 0$

- Hence, $\forall \varepsilon > 0$

- $\lim_{n \to \infty} \Pr \left[H(X_1) - \varepsilon \leq -\frac{1}{n} \log(p(X_1, \ldots, X_n)) \leq H(X_1) + \varepsilon \right] = 1$

Asymptotic equipartition theorem (AEP), cont.

- By weak law of large numbers:

  $\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathsf{E} \log p(X_1) = -H(X_1)$

- That is, $\lim_{n \to \infty} \Pr\left[\left|-\frac{1}{n} \log(p(X_1, \ldots, X_n)) - H(X_1)\right| > \varepsilon\right] = 0$, for any $\varepsilon > 0$

- Hence, $\forall \varepsilon > 0$

- $\lim_{n \to \infty} \Pr\left[H(X_1) - \varepsilon \leq -\frac{1}{n} \log(p(X_1, \ldots, X_n)) \leq H(X_1) + \varepsilon\right] = 1$

- $\lim_{n \to \infty} \Pr\left[2^{-H(X_1, \ldots, X_n) - \varepsilon n} \leq p(X_1, \ldots, X_n) \leq 2^{-H(X_1, \ldots, X_n) + \varepsilon n}\right] = 1$

Asymptotic equipartition theorem (AEP), cont.

- By weak law of large numbers:

  $\frac{1}{n} \log p(X_1, \ldots, X_n) = \frac{1}{n} \sum_i \log p(X_i) \xrightarrow{P} \mathrm{E} \log p(X_1) = -H(X_1)$

- That is, $\lim_{n \to \infty} \Pr\left[\left|-\frac{1}{n} \log(p(X_1, \ldots, X_n)) - H(X_1)\right| > \varepsilon\right] = 0$, for any $\varepsilon > 0$

- Hence, $\forall \varepsilon > 0$

- $\lim_{n \to \infty} \Pr\left[H(X_1) - \varepsilon \leq -\frac{1}{n} \log(p(X_1, \ldots, X_n)) \leq H(X_1) + \varepsilon\right] = 1$

- $\lim_{n \to \infty} \Pr\left[2^{-H(X_1, \ldots, X_n) - \varepsilon n} \leq p(X_1, \ldots, X_n) \leq 2^{-H(X_1, \ldots, X_n) + \varepsilon n}\right] = 1$

- What does it mean?

Typical values

Typical values

- Let $X_1, \ldots, X_n$ be iid $\sim p$

Typical values

- Let $X_1, \ldots, X_n$ be iid $\sim p$

- For $n \in \mathbb{N}$ and $\varepsilon > 0$, the typical sequence $A_{n,\varepsilon} :=$
  $\{(a_1, \ldots, a_n) \colon 2^{-n(H(X_1)+\varepsilon)} \leq \Pr[X_1 = a_1 \wedge \ldots \wedge X_n = a_n] \leq 2^{-n(H(X_1)-\varepsilon)}\}$

Typical values

- Let $X_1, \ldots, X_n$ be iid $\sim p$

- For $n \in \mathbb{N}$ and $\varepsilon > 0$, the typical sequence $A_{n,\varepsilon} :=$
  $\{(a_1, \ldots, a_n) \colon 2^{-n(H(X_1)+\varepsilon)} \leq \Pr[X_1 = a_1 \wedge \ldots \wedge X_n = a_n] \leq 2^{-n(H(X_1)-\varepsilon)}\}$

- $\frac{1}{2} \cdot 2^{n(H(X_1)-\varepsilon)} \leq |A_{n,\varepsilon}| \leq 2^{n(H(X_1)+\varepsilon)}$ (on board)

  (for the lower bound we assume $\Pr[(X_1, \ldots, X_n) \in A_{n,\varepsilon}] \geq \frac{1}{2}$)

Typical values

- Let $X_1, \ldots, X_n$ be iid $\sim p$

- For $n \in \mathbb{N}$ and $\varepsilon > 0$, the typical sequence $A_{n,\varepsilon} :=$
  $\{(a_1, \ldots, a_n) \colon 2^{-n(H(X_1)+\varepsilon)} \leq \Pr[X_1 = a_1 \wedge \ldots \wedge X_n = a_n] \leq 2^{-n(H(X_1)-\varepsilon)}\}$

- $\frac{1}{2} \cdot 2^{n(H(X_1)-\varepsilon)} \leq |A_{n,\varepsilon}| \leq 2^{n(H(X_1)+\varepsilon)}$ (on board)

  (for the lower bound we assume $\Pr[(X_1, \ldots, X_n) \in A_{n,\varepsilon}] \geq \frac{1}{2}$)

- Hence, $n(H(X_1) - \varepsilon) - 1 \leq \log|A_{n,\varepsilon}| \leq n(H(X_1) + \varepsilon)$

Typical values

- ▶ Let $X_1, \ldots, X_n$ be iid $\sim p$

- ▶ For $n \in \mathbb{N}$ and $\varepsilon > 0$, the typical sequence $A_{n,\varepsilon} :=$
  $\{(a_1, \ldots, a_n) \colon 2^{-n(H(X_1)+\varepsilon)} \leq \Pr[X_1 = a_1 \wedge \ldots \wedge X_n = a_n] \leq 2^{-n(H(X_1)-\varepsilon)}\}$

- ▶ $\frac{1}{2} \cdot 2^{n(H(X_1)-\varepsilon)} \leq |A_{n,\varepsilon}| \leq 2^{n(H(X_1)+\varepsilon)}$ (on board)

  (for the lower bound we assume $\Pr[(X_1, \ldots, X_n) \in A_{n,\varepsilon}] \geq \frac{1}{2}$)

- ▶ Hence, $n(H(X_1) - \varepsilon) - 1 \leq \log|A_{n,\varepsilon}| \leq n(H(X_1) + \varepsilon)$

- ▶ $\lim_{n \to \infty} \Pr[(X_1, \ldots, X_n) \notin A_{n,\varepsilon}] = 0$

Typical values

- Let $X_1, \ldots, X_n$ be iid $\sim p$

- For $n \in \mathbb{N}$ and $\varepsilon > 0$, the typical sequence $A_{n,\varepsilon} :=$
  $\{(a_1, \ldots, a_n) \colon 2^{-n(H(X_1)+\varepsilon)} \leq \Pr[X_1 = a_1 \wedge \ldots \wedge X_n = a_n] \leq 2^{-n(H(X_1)-\varepsilon)}\}$

- $\frac{1}{2} \cdot 2^{n(H(X_1)-\varepsilon)} \leq |A_{n,\varepsilon}| \leq 2^{n(H(X_1)+\varepsilon)}$ (on board)

  (for the lower bound we assume $\Pr[(X_1, \ldots, X_n) \in A_{n,\varepsilon}] \geq \frac{1}{2}$)

- Hence, $n(H(X_1) - \varepsilon) - 1 \leq \log |A_{n,\varepsilon}| \leq n(H(X_1) + \varepsilon)$

- $\lim_{n \to \infty} \Pr[(X_1, \ldots, X_n) \notin A_{n,\varepsilon}] = 0$

- So roughly, $(X_1, \ldots, X_n)$ is close to uniform over $A_{n,\varepsilon}$ and $|A_{n,\varepsilon}| \approx 2^{n(H(X_1))}$

Typical values

- Let $X_1, \ldots, X_n$ be iid $\sim p$

- For $n \in \mathbb{N}$ and $\varepsilon > 0$, the typical sequence $A_{n,\varepsilon} :=$
  $\{(a_1, \ldots, a_n) \colon 2^{-n(H(X_1)+\varepsilon)} \leq \Pr[X_1 = a_1 \wedge \ldots \wedge X_n = a_n] \leq 2^{-n(H(X_1)-\varepsilon)}\}$

- $\frac{1}{2} \cdot 2^{n(H(X_1)-\varepsilon)} \leq |A_{n,\varepsilon}| \leq 2^{n(H(X_1)+\varepsilon)}$ (on board)

  (for the lower bound we assume $\Pr[(X_1, \ldots, X_n) \in A_{n,\varepsilon}] \geq \frac{1}{2}$)

- Hence, $n(H(X_1) - \varepsilon) - 1 \leq \log |A_{n,\varepsilon}| \leq n(H(X_1) + \varepsilon)$

- $\lim_{n \to \infty} \Pr[(X_1, \ldots, X_n) \notin A_{n,\varepsilon}] = 0$

- So roughly, $(X_1, \ldots, X_n)$ is close to uniform over $A_{n,\varepsilon}$ and $|A_{n,\varepsilon}| \approx 2^{n(H(X_1)}$

- Recall that in statistical mechanics, entropy was define as the log (number of states the system can be at).

Typical values

- ▶ Let $X_1, \ldots, X_n$ be iid $\sim p$
- ▶ For $n \in \mathbb{N}$ and $\varepsilon > 0$, the typical sequence $A_{n,\varepsilon} :=$
  $\{(a_1, \ldots, a_n) \colon 2^{-n(H(X_1)+\varepsilon)} \le \Pr[X_1 = a_1 \wedge \ldots \wedge X_n = a_n] \le 2^{-n(H(X_1)-\varepsilon)}\}$
- ▶ $\frac{1}{2} \cdot 2^{n(H(X_1)-\varepsilon)} \le |A_{n,\varepsilon}| \le 2^{n(H(X_1)+\varepsilon)}$ (on board)

  (for the lower bound we assume $\Pr[(X_1, \ldots, X_n) \in A_{n,\varepsilon}] \ge \frac{1}{2}$)
- ▶ Hence, $n(H(X_1) - \varepsilon) - 1 \le \log |A_{n,\varepsilon}| \le n(H(X_1) + \varepsilon)$
- ▶ $\lim_{n\to\infty} \Pr[(X_1, \ldots, X_n) \notin A_{n,\varepsilon}] = 0$
- ▶ So roughly, $(X_1, \ldots, X_n)$ is close to uniform over $A_{n,\varepsilon}$ and $|A_{n,\varepsilon}| \approx 2^{n(H(X_1)}$

- ▶ Recall that in statistical mechanics, entropy was define as the log (number of states the system can be at).

- ▶ The above extends to many variables of different distributions, and not fully independent.

Part II

Data Compression

Data compression

Data compression

- Let $X_1, \ldots, X_n$ be iid $\sim p$

Data compression

- Let $X_1, \ldots, X_n$ be iid $\sim p$

- To describe $(X_1, \ldots, X_n)$ with negligible error, we need $H(X_1, \ldots, X_n) + \varepsilon n$ bits, where $\varepsilon \to 0$ as $n \leftarrow \infty$

Data compression

- Let $X_1, \ldots, X_n$ be iid $\sim p$

- To describe $(X_1, \ldots, X_n)$ with negligible error, we need $H(X_1, \ldots, X_n) + \varepsilon n$ bits, where $\varepsilon \to 0$ as $n \leftarrow \infty$

- So $H(X_1, \ldots, X_n)$ is approximately the number of bits it takes to describe $X_1, \ldots, X_n$

Lower bound

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

Lower bound

- Encoding function $f \colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function
  $g \colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)
- X rv over $\{0,1\}^n$, $Y = f(X)$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ \qquad (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function
  $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y)$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y)$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y) = I(X; Y)$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y) = I(X; Y) = H(Y) - H(Y|X)$

Lower bound

- Encoding function $f \colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g \colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y) = I(X;Y) = H(Y) - H(Y|X) \leq H(Y)$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y) = I(X; Y) = H(Y) - H(Y|X) \leq H(Y) \leq m$

Lower bound

- Encoding function $f: \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function
  $g: \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y) = I(X; Y) = H(Y) - H(Y|X) \leq H(Y) \leq m$

- Thus, $m \geq H(X) - \varepsilon n - 1$

Lower bound

- Encoding function $f\colon \{0,1\}^n \mapsto \{0,1\}^m$ and decoding function $g\colon \{0,1\}^m \mapsto \{0,1\}^n$ (typically $m < n$)

- X rv over $\{0,1\}^n$, $Y = f(X)$

- $X \to Y \to g(Y)$

- Assume $\Pr[g(Y) = X] \geq 1 - \varepsilon$ — g restores X w.h.p.

- By Fano, $H(X \mid Y)$ is small: $H(X|Y) \leq h(\varepsilon) + \varepsilon \log(2^n - 1) \leq \varepsilon n + 1$

- Hence,
  $H(X) - \varepsilon n - 1 \leq H(X) - H(X|Y) = I(X;Y) = H(Y) - H(Y|X) \leq H(Y) \leq m$

- Thus, $m \geq H(X) - \varepsilon n - 1$

- In case $H(X) = nH(X_1)$, then $m \geq n(H(X_1) - \varepsilon) - 1$

Codes

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C \colon \mathcal{X} \mapsto \Sigma^*$.

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C \colon \mathcal{X} \mapsto \Sigma^*$.

- We call $\{C(x) \colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C \colon \mathcal{X} \mapsto \Sigma^*$.

- We call $\{C(x) \colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)
- C is nonsingular, if it is injective over $\mathcal{X}$.

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C\colon \mathcal{X} \mapsto \Sigma^*$.

- We call $\{C(x)\colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)
- C is nonsingular, if it is injective over $\mathcal{X}$.
- For $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}^k$, let $C(\mathbf{x}) = C(x_1)C(x_2)\ldots C(x_k)$

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C\colon \mathcal{X} \mapsto \Sigma^*$.

- ▸ We call $\{C(x)\colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)
- ▸ C is nonsingular, if it is injective over $\mathcal{X}$.
- ▸ For $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}^k$, let $C(\mathbf{x}) = C(x_1)C(x_2)\ldots C(x_k)$
- ▸ C is uniquely decodable, if it is nonsingular over $\mathcal{X}^*$

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C \colon \mathcal{X} \mapsto \Sigma^*$.

- ▶ We call $\{C(x) \colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)
- ▶ C is nonsingular, if it is injective over $\mathcal{X}$.
- ▶ For $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}^k$, let $C(\mathbf{x}) = C(x_1)C(x_2)\ldots C(x_k)$
- ▶ C is uniquely decodable, if it is nonsingular over $\mathcal{X}^*$
- ▶ Uniquely decodable $\implies$ nonsingular   (other direction is not true)

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C \colon \mathcal{X} \mapsto \Sigma^*$.

- ▶ We call $\{C(x) \colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)
- ▶ C is nonsingular, if it is injective over $\mathcal{X}$.
- ▶ For $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}^k$, let $C(\mathbf{x}) = C(x_1)C(x_2)\ldots C(x_k)$
- ▶ C is uniquely decodable, if it is nonsingular over $\mathcal{X}^*$
- ▶ Uniquely decodable $\implies$ nonsingular   (other direction is not true)
- ▶ A code is prefix code (or instantaneous code), if no code word is a prefix of another code word

Codes

> **Definition 1 (Codes)**
>
> A code for random variable X over $\mathcal{X}$ is a mapping $C\colon \mathcal{X} \mapsto \Sigma^*$.

- ▶ We call $\{C(x)\colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)
- ▶ C is nonsingular, if it is injective over $\mathcal{X}$.
- ▶ For $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}^k$, let $C(\mathbf{x}) = C(x_1)C(x_2)\ldots C(x_k)$
- ▶ C is uniquely decodable, if it is nonsingular over $\mathcal{X}^*$
- ▶ Uniquely decodable $\implies$ nonsingular   (other direction is not true)
- ▶ A code is prefix code (or instantaneous code), if no code word is a prefix of another code word
- ▶ Prefix code $\implies$ uniquely decodable

Codes

Definition 1 (Codes)

A code for random variable X over $\mathcal{X}$ is a mapping $C\colon \mathcal{X} \mapsto \Sigma^*$.

- ► We call $\{C(x)\colon x \in \mathcal{X}\}$ the codewords of C (with respect to X)
- ► C is nonsingular, if it is injective over $\mathcal{X}$.
- ► For $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}^k$, let $C(\mathbf{x}) = C(x_1)C(x_2)\ldots C(x_k)$
- ► C is uniquely decodable, if it is nonsingular over $\mathcal{X}^*$
- ► Uniquely decodable $\implies$ nonsingular (other direction is not true)
- ► A code is prefix code (or instantaneous code), if no code word is a prefix of another code word
- ► Prefix code $\implies$ uniquely decodable
- ► We focus on binary prefix codes ($\Sigma = \{0, 1\}$)

Examples

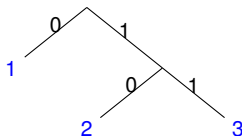- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$).

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i)$).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
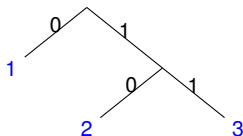- The code

Examples

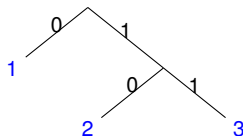- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
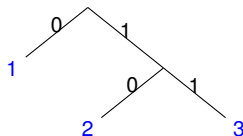- The code

x	$C(x)$
1	0
2	10
3	11

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

x	$C(x)$
1	0
2	10
3	11

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i)$).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

x	$C(x)$
1	0
2	10
3	11

- Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

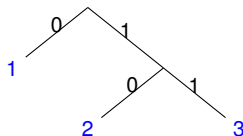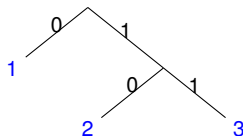| x	$C(x)$
1	0
2	10
3	11

- Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$
- $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

x	$C(x)$
1	0
2	10
3	11

- Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$
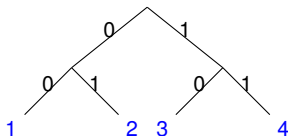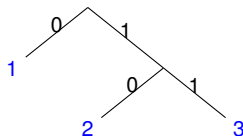- $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

x	$C(x)$
1	0
2	10
3	11

- Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$
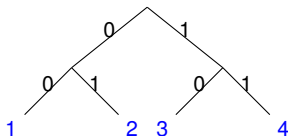- $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

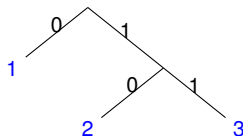| x	$C(x)$
1	0
2	10
3	11

- Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$
- $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

x	$C(x)$
1	0
2	10
3	11

- Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$
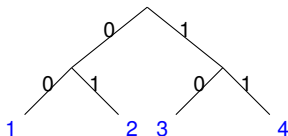- $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$

Examples

- $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i$)).
- We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- The code

x	$C(x)$
1	0
2	10
3	11

- Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$
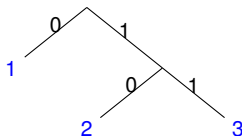- $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$
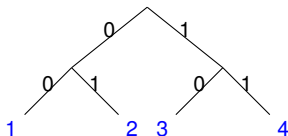

Or

Examples

- ▶ $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i)$).
- ▶ We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$
- ▶ The code

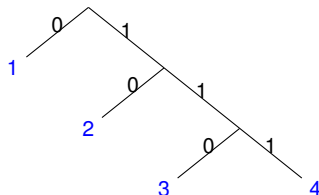| x	$C(x)$
1	0
2	10
3	11

- ▶ Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$
- ▶ $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$

Or

Examples

▶ $X \sim (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ (i.e., $\Pr[X = i] = p_i)$).

▶ We can use one bit to tel whether $X = 1$ or $X \in \{2, 3\}$, and another bit to tell whether $X = 2$ or $X = 3$

▶ The code

x	$C(x)$
1	0
2	10
3	11

▶ Expected encoding length: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1\frac{1}{2}$

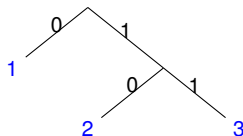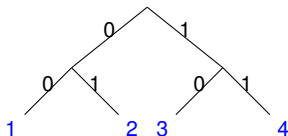▶ $X \sim (\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$

Or

▶ All are prefix codes: no codeword is a prefix of another codeword

Prefix codes

Prefix codes

- Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))

Prefix codes

- Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))
- We want to place $\{1, \ldots, m\}$ on the leaves of a binary tree T (not necessarily in order):

Prefix codes

- ▶ Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))

- ▶ We want to place $\{1, \ldots, m\}$ on the leaves of a binary tree T (not necessarily in order):
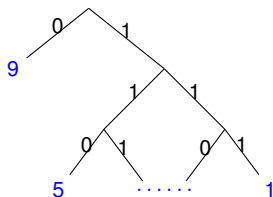
Prefix codes

- Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))
- We want to place $\{1, \ldots, m\}$ on the leaves of a binary tree T (not necessarily in order):

- Every symbol is encoded by the bits on the path leading to it.

Prefix codes

- Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))

- We want to place $\{1, \ldots, m\}$ on the leaves of a binary tree T (not necessarily in order):

- Every symbol is encoded by the bits on the path leading to it.

- This yields a binary prefix code.

Prefix codes

▶ Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))

▶ We want to place $\{1, \ldots, m\}$ on the leaves of a binary tree T (not necessarily in order):

▶ Every symbol is encoded by the bits on the path leading to it.

▶ This yields a binary prefix code.

▶ Every prefix code can be represented as such a tree

Prefix codes

- Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))

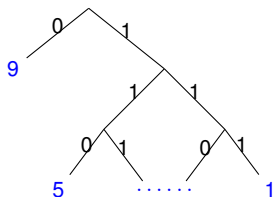- We want to place $\{1, \ldots, m\}$ on the leaves of a binary tree T (not necessarily in order):

- Every symbol is encoded by the bits on the path leading to it.

- This yields a binary prefix code.

- Every prefix code can be represented as such a tree

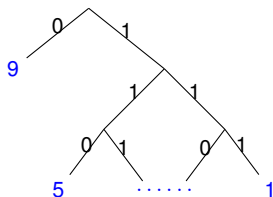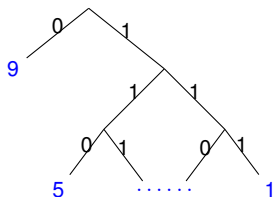- We identify prefix codes with their trees.

Prefix codes

- ► Let $X \sim (p_1, \ldots, p_m)$ (i.e., $\Pr[X = i] = p_i$))
- ► We want to place $\{1, \ldots, m\}$ on the leaves of a binary tree T (not necessarily in order):
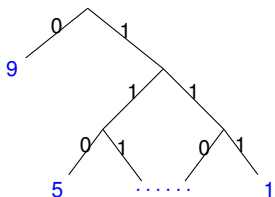
- ► Every symbol is encoded by the bits on the path leading to it.
- ► This yields a binary prefix code.
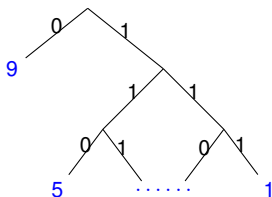- ► Every prefix code can be represented as such a tree
- ► We identify prefix codes with their trees.
- ► Encoding/decoding is clear (and highly efficient)

Code length

Code length

- For a prefix code C over X, let $\ell(x) = |C(x)|$ (i.e., # of bits in x)

Code length

- For a prefix code C over X, let $\ell(x) = |C(x)|$ (i.e., # of bits in x)
- Since C a prefix code, $\ell(x)$ is the depth of x in the code tree of C

Code length

- For a prefix code C over X, let $\ell(x) = |C(x)|$ (i.e., # of bits in x)

- Since C a prefix code, $\ell(x)$ is the depth of x in the code tree of C

- $L(C) := E(\ell(X))$ is the average code length (of C with respect to X)

Code length

- For a prefix code C over X, let $\ell(x) = |C(x)|$ (i.e., # of bits in x)

- Since C a prefix code, $\ell(x)$ is the depth of x in the code tree of C

- $L(C) := \mathsf{E}(\ell(X))$ is the average code length (of C with respect to X)

- We sometimes speak about $L(T)$ where T is the tree representation of C

Code length

- For a prefix code C over X, let $\ell(x) = |C(x)|$ (i.e., # of bits in x)

- Since C a prefix code, $\ell(x)$ is the depth of x in the code tree of C

- $L(C) := E(\ell(X))$ is the average code length (of C with respect to X)

- We sometimes speak about $L(T)$ where T is the tree representation of C

- $L(X)$ is the code length of the optimal prefix code for X

Code length

- For a prefix code C over X, let $\ell(x) = |C(x)|$ (i.e., # of bits in x)

- Since C a prefix code, $\ell(x)$ is the depth of x in the code tree of C

- $L(C) := E(\ell(X))$ is the average code length (of C with respect to X)

- We sometimes speak about $L(T)$ where T is the tree representation of C

- $L(X)$ is the code length of the optimal prefix code for X

- How small can $L(X)$ be?

Code length

- ► For a prefix code C over X, let $\ell(x) = |C(x)|$ (i.e., # of bits in x)

- ► Since C a prefix code, $\ell(x)$ is the depth of x in the code tree of C

- ► $L(C) := E(\ell(X))$ is the average code length (of C with respect to X)

- ► We sometimes speak about $L(T)$ where T is the tree representation of C

- ► $L(X)$ is the code length of the optimal prefix code for X

- ► How small can $L(X)$ be?

- ► It turns out that $H(X) \leq L(X) \leq H(X) + 1$!

Huffman code

Huffman code

- ► Story...

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)

Huffman code

- Story...
- Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)

Huffman code

- Story...
- Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m - 1$ and m

Huffman code

- ► Story...
- ► Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ► Let v be (one of) the deepest vertex in T
- ► wlg. the descendants of v are $m-1$ and m

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m (otherwise, we can force it w/o increasing $L(T)$)

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m (otherwise, we can force it w/o increasing $L(T)$)

  - ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

- ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
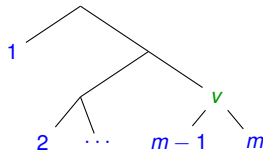- ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m (otherwise, we can force it w/o increasing $L(T)$)

- ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
- ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
- ▶ T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
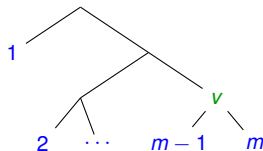
Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

- ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
- ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
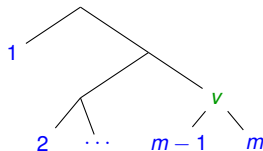- ▶ T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.

Huffman code

- ► Story...
- ► Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ► Let v be (one of) the deepest vertex in T
- ► wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

  - ► T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
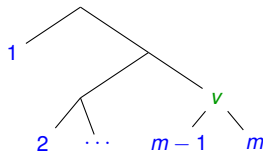 - ► $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
  - ► T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
    (otherwise, we can improve T' and hence improve T)

Huffman code

- ► Story...
- ► Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ► Let v be (one of) the deepest vertex in T
- ► wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

  - ► T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
  - ► $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
  - ► T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
    (otherwise, we can improve T' and hence improve T)
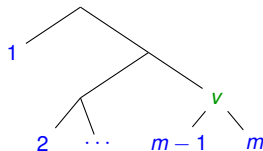- ► Huffman algorithm:

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

- ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
- ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
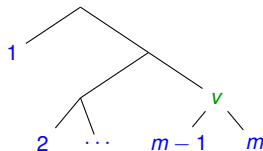- ▶ T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
  (otherwise, we can improve T' and hence improve T)
- ▶ Huffman algorithm:
  1. Sort $p_1, \ldots, p_m$

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

  - ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
  - ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
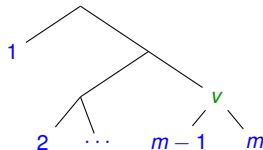 - ▶ T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
    (otherwise, we can improve T' and hence improve T)
  - ▶ Huffman algorithm:
    1. Sort $p_1, \ldots, p_m$
    2. Find (via recursions) the best tree for $(p_1, \ldots, p_{m-1} + p_m)$

Huffman code

- ► Story...
- ► Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ► Let v be (one of) the deepest vertex in T
- ► wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

- - ► T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
  - ► $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
  - ► T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
    (otherwise, we can improve T' and hence improve T)
- ► Huffman algorithm:
  1. Sort $p_1, \ldots, p_m$
  2. Find (via recursions) the best tree for $(p_1, \ldots, p_{m-1} + p_m)$
  3. Replace leaf $\{m-1, m\}$ with the depth-one tree of leaves $m-1, m$
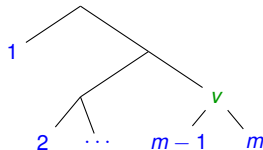
Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

  - ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
  - ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
  - ▶ T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
    (otherwise, we can improve T' and hence improve T)
- ▶ Huffman algorithm:
  1. Sort $p_1, \ldots, p_m$
  2. Find (via recursions) the best tree for $(p_1, \ldots, p_{m-1} + p_m)$
  3. Replace leaf $\{m-1, m\}$ with the depth-one tree of leaves $m-1, m$
- ▶ Huffman is an optimal binary prefix code.
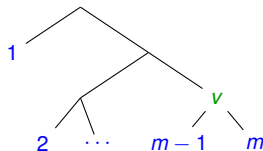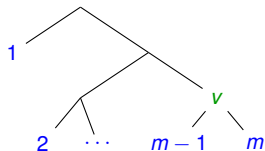
Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

  - ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
  - ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
  - ▶ T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
    (otherwise, we can improve T' and hence improve T)
- ▶ Huffman algorithm:
  1. Sort $p_1, \ldots, p_m$
  2. Find (via recursions) the best tree for $(p_1, \ldots, p_{m-1} + p_m)$
  3. Replace leaf $\{m-1, m\}$ with the depth-one tree of leaves $m-1, m$
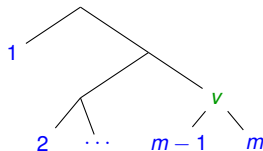- ▶ Huffman is an optimal binary prefix code.

Huffman code

- ▶ Story...
- ▶ Suppose T is optimal tree for $X \sim (p_1, \ldots, p_m)$ (wlg. $p_1 \geq p_2 \geq \ldots \geq p_m$)
- ▶ Let v be (one of) the deepest vertex in T
- ▶ wlg. the descendants of v are $m-1$ and m
  (otherwise, we can force it w/o increasing $L(T)$)

- ▶ T' – generated from T be replacing the sub-tree rooted in v with the symbol $\{m-1, m\}$
- ▶ $L(T) = L(T') + (p_{m-1} + p_m) \cdot 1$
- ▶ T' is optimal tree for $X' \sim (p_1, \ldots, p_{m-1} + p_m)$.
  (otherwise, we can improve T' and hence improve T)
- ▶ Huffman algorithm:
  1. Sort $p_1, \ldots, p_m$
  2. Find (via recursions) the best tree for $(p_1, \ldots, p_{m-1} + p_m)$
  3. Replace leaf $\{m-1, m\}$ with the depth-one tree of leaves $m-1, m$
- ▶ Huffman is an optimal binary prefix code. Proof: ?

Huffman code, example

Huffman code, example

- $X \sim (.25, .25, .2, .15, .15)$

Huffman code, example
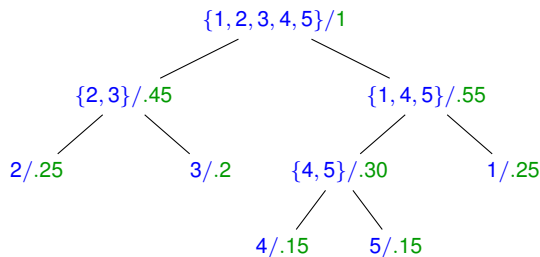
- $X \sim (.25, .25, .2, .15, .15)$

Huffman code, example

▸ $X \sim (.25, .25, .2, .15, .15)$
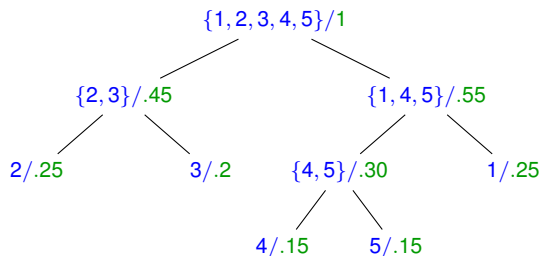
▸ On board...

Craft inequality

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Theorem extends to the countably infinite case (not proven here).

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Theorem extends to the countably infinite case (not proven here).

First part:

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Theorem extends to the countably infinite case (not proven here).

First part:

- ▶ Denote the i'th codeword by i

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Theorem extends to the countably infinite case (not proven here).

First part:

- Denote the i'th codeword by i
- Let Y the leaf reached by a uniform random walk on the code tree

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Theorem extends to the countably infinite case (not proven here).

First part:

- Denote the i'th codeword by i
- Let Y the leaf reached by a uniform random walk on the code tree
- $\Pr[Y = i] = 2^{-\ell_i}$.

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Theorem extends to the countably infinite case (not proven here).

First part:

- Denote the i'th codeword by i
- Let Y the leaf reached by a uniform random walk on the code tree
- $\Pr[Y = i] = 2^{-\ell_i}$.
- Hence, $\sum_i 2^{-\ell_i} = \sum_i \Pr[Y = i] \leq 1$

Craft inequality

> **Theorem 2 (Craft inequality)**
>
> *Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy*
> $$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$
> *Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.*

Theorem extends to the countably infinite case (not proven here).

First part:

- Denote the i'th codeword by i
- Let Y the leaf reached by a uniform random walk on the code tree
- $\Pr[Y = i] = 2^{-\ell_i}$.
- Hence, $\sum_i 2^{-\ell_i} = \sum_i \Pr[Y = i] \leq 1$

Craft inequality

Theorem 2 (Craft inequality)

Let C be (binary) prefix code. Then its codewords lengths $\ell_1, \ldots, \ell_m$ satisfy

$$\sum_{i \in [m]} 2^{-\ell_i} \leq 1.$$

Conversely, for any $\ell_1, \ldots, \ell_m$ satisfying the inequality, there exists a prefix code with these lengths.

Theorem extends to the countably infinite case (not proven here).

First part:

- Denote the i'th codeword by i
- Let Y the leaf reached by a uniform random walk on the code tree
- $\Pr[Y = i] = 2^{-\ell_i}$.
- Hence, $\sum_i 2^{-\ell_i} = \sum_i \Pr[Y = i] \leq 1$

For finite code, proof can be carried using simple induction on code tree depth.

Craft inequality. cont.

Craft inequality. cont.

- ▶ Let $\ell_1 \le \ldots \le \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \le 1$

Craft inequality. cont.

- ► Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$
- ► We construct a tree of m codewords with the above lengths.

Craft inequality. cont.

- ► Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$
- ► We construct a tree of m codewords with the above lengths.

    1. Start with a binary tree of depth ℓ_m

Craft inequality. cont.

- ► Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

- ► We construct a tree of m codewords with the above lengths.

  1. Start with a binary tree of depth ℓ_m

  2. At step i, assign the first unassigned node of depth ℓ_i to the i'th codeword, and remove its descendants from the tree.

Craft inequality. cont.

- ▶ Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$
- ▶ We construct a tree of m codewords with the above lengths.
    1. Start with a binary tree of depth ℓ_m
    2. At step i, assign the first unassigned node of depth ℓ_i to the i'th codeword, and remove its descendants from the tree.
- ▶ If completed, the algorithm yields the desired code.

Craft inequality. cont.

- ► Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

- ► We construct a tree of m codewords with the above lengths.

    1. Start with a binary tree of depth ℓ_m
    2. At step i, assign the first unassigned node of depth ℓ_i to the i'th codeword, and remove its descendants from the tree.

- ► If completed, the algorithm yields the desired code.
- ► Claim: the algorithm always completes.

Craft inequality. cont.

▶ Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

▶ We construct a tree of m codewords with the above lengths.

    1. Start with a binary tree of depth ℓ_m

    2. At step i, assign the first unassigned node of depth ℓ_i to the i'th codeword, and remove its descendants from the tree.

▶ If completed, the algorithm yields the desired code.

▶ Claim: the algorithm always completes.

    ▶ Let $\mathcal{S}_\ell(i)$ be the nodes of depth $\ell \geq \ell_i$ that made unavailable when assigning a node to i'th codeword

Craft inequality. cont.

- ► Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

- ► We construct a tree of *m* codewords with the above lengths.

    1. Start with a binary tree of depth ℓ_m
    2. At step *i*, assign the first unassigned node of depth ℓ_i to the *i*'th codeword, and remove its descendants from the tree.

- ► If completed, the algorithm yields the desired code.

- ► Claim: the algorithm always completes.

    - ► Let $\mathcal{S}_\ell(i)$ be the nodes of depth $\ell \geq \ell_i$ that made unavailable when assigning a node to *i*'th codeword
    - ► $|\mathcal{S}_\ell(i)| = 2^{\ell - \ell_i}$

Craft inequality. cont.

- ► Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

- ► We construct a tree of *m* codewords with the above lengths.

  1. Start with a binary tree of depth ℓ_m
  2. At step *i*, assign the first unassigned node of depth ℓ_i to the *i*'th codeword, and remove its descendants from the tree.

- ► If completed, the algorithm yields the desired code.

- ► Claim: the algorithm always completes.

  - ► Let $\mathcal{S}_\ell(i)$ be the nodes of depth $\ell \geq \ell_i$ that made unavailable when assigning a node to *i*'th codeword
  - ► $|\mathcal{S}_\ell(i)| = 2^{\ell - \ell_i}$
  - ► $2^{-\ell_i} = |\mathcal{S}_\ell(i)| \cdot 2^{-\ell}$

Craft inequality. cont.

- ▶ Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

- ▶ We construct a tree of *m* codewords with the above lengths.

    1. Start with a binary tree of depth ℓ_m
    2. At step *i*, assign the first unassigned node of depth ℓ_i to the *i*'th codeword, and remove its descendants from the tree.

- ▶ If completed, the algorithm yields the desired code.

- ▶ Claim: the algorithm always completes.

    - ▶ Let $\mathcal{S}_\ell(i)$ be the nodes of depth $\ell \geq \ell_i$ that made unavailable when assigning a node to *i*'th codeword
    - ▶ $|\mathcal{S}_\ell(i)| = 2^{\ell - \ell_i}$
    - ▶ $2^{-\ell_i} = |\mathcal{S}_\ell(i)| \cdot 2^{-\ell}$
    - ▶ Let $\hat{\mathcal{S}}_\ell(i) = \bigcup_{j=1}^{i-1} \mathcal{S}_\ell(j)$ — the nodes of depth ℓ unavailable at the beginning of step *i*

Craft inequality. cont.

- Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

- We construct a tree of *m* codewords with the above lengths.

    1. Start with a binary tree of depth ℓ_m
    2. At step *i*, assign the first unassigned node of depth ℓ_i to the *i*'th codeword, and remove its descendants from the tree.

- If completed, the algorithm yields the desired code.

- Claim: the algorithm always completes.

    - Let $\mathcal{S}_\ell(i)$ be the nodes of depth $\ell \geq \ell_i$ that made unavailable when assigning a node to *i*'th codeword
    - $|\mathcal{S}_\ell(i)| = 2^{\ell - \ell_i}$
    - $2^{-\ell_i} = |\mathcal{S}_\ell(i)| \cdot 2^{-\ell}$
    - Let $\hat{\mathcal{S}}_\ell(i) = \bigcup_{j=1}^{i-1} \mathcal{S}_\ell(j)$ — the nodes of depth ℓ unavailable at the beginning of step *i*
    - $\sum_{j \in [i-1]} 2^{-|\ell_j|} = \left| \hat{\mathcal{S}}_\ell(i) \right| \cdot 2^{-\ell}$

Craft inequality. cont.

- Let $\ell_1 \leq \ldots \leq \ell_m$ be such that $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

- We construct a tree of m codewords with the above lengths.

  1. Start with a binary tree of depth ℓ_m
  2. At step i, assign the first unassigned node of depth ℓ_i to the i'th codeword, and remove its descendants from the tree.

- If completed, the algorithm yields the desired code.

- Claim: the algorithm always completes.

  - Let $\mathcal{S}_\ell(i)$ be the nodes of depth $\ell \geq \ell_i$ that made unavailable when assigning a node to i'th codeword
  - $|\mathcal{S}_\ell(i)| = 2^{\ell - \ell_i}$
  - $2^{-\ell_i} = |\mathcal{S}_\ell(i)| \cdot 2^{-\ell}$
  - Let $\hat{\mathcal{S}}_\ell(i) = \bigcup_{j=1}^{i-1} \mathcal{S}_\ell(j)$ — the nodes of depth ℓ unavailable at the beginning of step i
  - $\sum_{j \in [i-1]} 2^{-|\ell_j|} = \left| \hat{\mathcal{S}}_\ell(i) \right| \cdot 2^{-\ell}$
  - Hence, at beginning of step i exists available depth-ℓ_i node.

Optimal code

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with $H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $- \sum_{i \in [m]} p_i \log p_i \leq - \sum p_i \log q_i = \sum_i p_i \ell_i$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \le L(C) \le H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \le -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \le L(C) \le H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \le -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \le L(C)$.

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \le L(C) \le H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \le -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \le L(C)$.

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \leq L(C)$.

Proving upper bound:

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$$H(X) \leq L(C) \leq H(X) + 1$$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \leq L(C)$.

Proving upper bound:

- $\ell_i = \lceil -\log p_i \rceil$.

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \leq L(C)$.

Proving upper bound:

- $\ell_i = \lceil -\log p_i \rceil$.
- $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \le L(C) \le H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \le -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \le L(C)$.

Proving upper bound:

- $\ell_i = \lceil -\log p_i \rceil$.
- $\sum_{i \in [m]} 2^{-\ell_i} \le 1$
- There exists a (boolean prefix) code C for X with $C(i) = \ell_i$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \le L(C) \le H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \le -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \le L(C)$.

Proving upper bound:

- $\ell_i = \lceil -\log p_i \rceil$.
- $\sum_{i \in [m]} 2^{-\ell_i} \le 1$
- There exists a (boolean prefix) code C for X with $C(i) = \ell_i$
- $L(C) = \sum_i p_i \ell_i$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- ▶ Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- ▶ Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- ▶ By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- ▶ Hence $H(X) \leq L(C)$.

Proving upper bound:

- ▶ $\ell_i = \lceil -\log p_i \rceil$.
- ▶ $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$
- ▶ There exists a (boolean prefix) code C for X with $C(i) = \ell_i$
- ▶ $L(C) = \sum_i p_i \ell_i$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- Hence $H(X) \leq L(C)$.

Proving upper bound:

- $\ell_i = \lceil -\log p_i \rceil$.
- $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$
- There exists a (boolean prefix) code C for X with $C(i) = \ell_i$
- $L(C) = \sum_i p_i \ell_i \leq \sum_i p_i (-\log p_i + 1)$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

- ▶ Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).
- ▶ Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.
- ▶ By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$
- ▶ Hence $H(X) \leq L(C)$.

Proving upper bound:

- ▶ $\ell_i = \lceil -\log p_i \rceil$.
- ▶ $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$
- ▶ There exists a (boolean prefix) code C for X with $C(i) = \ell_i$
- ▶ $L(C) = \sum_i p_i \ell_i \leq \sum_i p_i (-\log p_i + 1) = -\sum_i p_i \log p_i + \sum_i p_i$

Optimal code

Theorem 3

For any rv X, there exists a prefix binary code C with
$H(X) \leq L(C) \leq H(X) + 1$

Proving lower bound:

▶ Let C be a binary prefix code for $X \sim p = (p_1, \ldots, p_m)$, and let $\ell_i = |C(i)|$. (As usual, we assume wlg. that $p_i = \Pr[X = i]$).

▶ Let $q = (q_1 = 2^{-\ell_1}, \ldots, q_m = 2^{-\ell_m}, q_{m+1} = 1 - \sum_{i \in [m]} q_i)$.

▶ By Jensen, $-\sum_{i \in [m]} p_i \log p_i \leq -\sum p_i \log q_i = \sum_i p_i \ell_i = L(C)$

▶ Hence $H(X) \leq L(C)$.

Proving upper bound:

▶ $\ell_i = \lceil -\log p_i \rceil$.

▶ $\sum_{i \in [m]} 2^{-\ell_i} \leq 1$

▶ There exists a (boolean prefix) code C for X with $C(i) = \ell_i$

▶ $L(C) = \sum_i p_i \ell_i \leq \sum_i p_i (-\log p_i + 1) = -\sum_i p_i \log p_i + \sum_i p_i = H(X) + 1$

Discrete distribution generation

Discrete distribution generation

Definition 4

Algorithm A generates the rv $X \sim \{p_1, \ldots, p_m\}$. if the following holds: in each step, A either stops or flips a coin $\sim (q_i, 1 - q_i)$.[a] After it stop, A outputs a value in $\mathbb{N}$. The probability that A outputs i is p_i.

___
[a] q_i can be a function of previous coin outcomes.

Discrete distribution generation

Definition 4

Algorithm A generates the rv $X \sim \{p_1, \ldots, p_m\}$. if the following holds: in each step, A either stops or flips a coin $\sim (q_i, 1 - q_i)$.[a] After it stop, A outputs a value in $\mathbb{N}$. The probability that A outputs i is p_i.

---

[a] q_i can be a function of previous coin outcomes.

Proposition 5

Let X be rv, and let G be the expected number of coins used by its best generating algorithm. Then $H(X) \leq G(X) \leq H(X) + 1$. If each p_i is a power of 2 (i.e., 2^{-k} for some $k \in \mathbb{Z}$), then $G(X) = X$.

Discrete distribution generation

Definition 4

Algorithm A generates the rv $X \sim \{p_1, \ldots, p_m\}$. if the following holds: in each step, A either stops or flips a coin $\sim (q_i, 1 - q_i)$.[a] After it stop, A outputs a value in $\mathbb{N}$. The probability that A outputs i is p_i.

---

[a] q_i can be a function of previous coin outcomes.

Proposition 5

Let X be rv, and let G be the expected number of coins used by its best generating algorithm. Then $H(X) \leq G(X) \leq H(X) + 1$. If each p_i is a power of 2 (i.e., 2^{-k} for some $k \in \mathbb{Z}$), then $G(X) = X$.

Proof: ?

Discrete distribution generation

Definition 4

Algorithm A generates the rv $X \sim \{p_1, \ldots, p_m\}$. if the following holds: in each step, A either stops or flips a coin $\sim (q_i, 1 - q_i)$.[a] After it stop, A outputs a value in $\mathbb{N}$. The probability that A outputs i is p_i.

---
[a] q_i can be a function of previous coin outcomes.

Proposition 5

Let X be rv, and let G be the expected number of coins used by its best generating algorithm. Then $H(X) \leq G(X) \leq H(X) + 1$. If each p_i is a power of 2 (i.e., 2^{-k} for some $k \in \mathbb{Z}$), then $G(X) = X$.

Proof: ? HW

Discrete distribution generation

Definition 4

Algorithm A generates the rv $X \sim \{p_1, \ldots, p_m\}$. if the following holds: in each step, A either stops or flips a coin $\sim (q_i, 1 - q_i)$.[a] After it stop, A outputs a value in $\mathbb{N}$. The probability that A outputs i is p_i.

---

[a] q_i can be a function of previous coin outcomes.

Proposition 5

Let X be rv, and let G be the expected number of coins used by its best generating algorithm. Then $H(X) \leq G(X) \leq H(X) + 1$. If each p_i is a power of 2 (i.e., 2^{-k} for some $k \in \mathbb{Z}$), then $G(X) = X$.

Proof: ? HW

Proposition 6

Let X be a rv, and let G_b be the expected number of coins used by its best generating algorithm that only flips uniform coins. Then
$H(X) \leq G_b(X) \leq H(X) + 2$.

Proving Proposition 6

Proving Proposition 6

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

Proving Proposition 6

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(M(i)) = -\log p_i$.

Proving Proposition 6

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(M(i)) = -\log p_i$.

- A uniform random walk on T, starting from the root, generates X

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- A uniform random walk on T, starting from the root, generates X

- Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- ▶ By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- ▶ A uniform random walk on T, starting from the root, generates X

- ▶ Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- ▶ By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- ▶ A uniform random walk on T, starting from the root, generates X

- ▶ Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- ▶ By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- ▶ A uniform random walk on T, starting from the root, generates X

- ▶ Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- ▶ Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- ► By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- ► A uniform random walk on T, starting from the root, generates X

- ► Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- ► Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$

- ► Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- ▶ By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- ▶ A uniform random walk on T, starting from the root, generates X

- ▶ Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- ▶ Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$

- ▶ Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$

- ▶ Let $Y = (X, K_X)$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- ▶ By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- ▶ A uniform random walk on T, starting from the root, generates X

- ▶ Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- ▶ Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$

- ▶ Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$

- ▶ Let $Y = (X, K_X)$

- ▶ $\Pr[Y = (i,j)] = p_i^{(j)}$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

▶ By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

▶ A uniform random walk on T, starting from the root, generates X

▶ Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

▶ Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$

▶ Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$

▶ Let $Y = (X, K_X)$

▶ $\Pr[Y = (i, j)] = p_i^{(j)}$

▶ $G_b(X) \leq G(Y)$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.
- A uniform random walk on T, starting from the root, generates X
- Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$
- Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$
- Let $Y = (X, K_X)$
- $\Pr[Y = (i,j)] = p_i^{(j)}$
- $G_b(X) \leq G(Y)$

Proving **Proposition 6**

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.

- A uniform random walk on T, starting from the root, generates X

- Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$

- Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$

- Let $Y = (X, K_X)$

- $\Pr[Y = (i,j)] = p_i^{(j)}$

- $G_b(X) \leq G(Y) = G_b(Y)$

Proving Proposition 6

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(M(i)) = -\log p_i$.
- A uniform random walk on T, starting from the root, generates X
- Expected number of coins used is $\sum_i p_i \ell(M(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$
- Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$
- Let $Y = (X, K_X)$
- $\Pr[Y = (i,j)] = p_i^{(j)}$
- $G_b(X) \leq G(Y) = G_b(Y) = H(Y)$

Proving Proposition 6

Let $X \sim \{p_1, p_2, \ldots\}$ be such that each p_i is a power of 2.

- By extended Craft inequality, exists an (infinite) binary tree T and mapping M from $\mathbb{N}$ to its leaves, such that $\ell(\mathsf{M}(i)) = -\log p_i$.
- A uniform random walk on T, starting from the root, generates X
- Expected number of coins used is $\sum_i p_i \ell(\mathsf{M}(i)) = -\sum_i p_i \log p_i = H(X)$

Let $X \sim \{p_1, \ldots, p_n\}$

- Let $(p_{i,1}, p_{i,2}, \ldots)$ be the binary representation of p_i and let $p_i^{(j)} = p_{i,j} \cdot 2^{-j}$
- Define K_i over $\mathbb{N}$ by $\Pr[K_i = j] = \frac{p_i^{(j)}}{p_i}$
- Let $Y = (X, K_X)$
- $\Pr[Y = (i, j)] = p_i^{(j)}$
- $G_b(X) \leq G(Y) = G_b(Y) = H(Y)$
- We conclude the proof showing that $H(Y) \leq H(X) + 2$.

Proving $H(Y) \leq H(X) + 2$

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$

Proving $H(Y) \leq H(X) + 2$

▶ Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$

▶ Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

- Tight proof:

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

- Tight proof:

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

- Tight proof:

$$H(Y) = -\sum_{i \in [m]} \sum_{j \in \mathbb{N}} p_i^{(j)} \log p_i^{(j)}$$

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

- Tight proof:

$$H(Y) \;\; = \;\; -\sum_{i \in [m]} \sum_{j \in \mathbb{N}} p_i^{(j)} \log p_i^{(j)} = \sum_i \sum_{j \,:\, p_i^{(j)} > 0} j \cdot 2^{-j}$$

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

- Tight proof:

$$H(Y) \;\; = \;\; -\sum_{i \in [m]} \sum_{j \in \mathbb{N}} p_i^{(j)} \log p_i^{(j)} = \sum_i \sum_{j:\, p_i^{(j)} > 0} j \cdot 2^{-j}$$

- Claim: $T_i := \sum_{j:\, p_i^{(j)} > 0} j \cdot 2^{-j} \leq -p_i \log p_i + 2p_i$.

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

- Tight proof:

$$H(Y) \;=\; -\sum_{i \in [m]} \sum_{j \in \mathbb{N}} p_i^{(j)} \log p_i^{(j)} = \sum_i \sum_{j : \, p_i^{(j)} > 0} j \cdot 2^{-j}$$

- Claim: $T_i := \sum_{j : \, p_i^{(j)} > 0} j \cdot 2^{-j} \leq -p_i \log p_i + 2p_i$.

- Proof: ?

Proving $H(Y) \leq H(X) + 2$

- Since $H(Y) = H(Y, X) = H(X) + H(Y|X)$, the proof is immediate if each p_i is of the form $(0, 0, \ldots, 0, 1, 1, 1, \ldots)$ ($Z \sim G(q)$ then $h(Z) = \frac{h(q)}{q}$)

- A simple reduction yields that $H(Y|X) < 2/\frac{1}{2} = 4$

- Tight proof:

$$H(Y) = -\sum_{i \in [m]} \sum_{j \in \mathbb{N}} p_i^{(j)} \log p_i^{(j)} = \sum_i \sum_{j \,:\, p_i^{(j)} > 0} j \cdot 2^{-j}$$

- Claim: $T_i := \sum_{j \,:\, p_i^{(j)} > 0} j \cdot 2^{-j} \leq -p_i \log p_i + 2p_i$.

- Proof: ?

- Hence, $H(Y) = \sum_i T_i \leq -\sum_i -p_i \log p_i + 2 \sum_i p_i = H(X) + 2$

Part III

Gambling

Horse Racing

Horse Racing

- Horses $\{1, \ldots, m\}$

Horse Racing

- ▶ Horses $\{1, \ldots, m\}$
- ▶ If horse i wins, gambler get payoff o_i per 1

Horse Racing

- Horses $\{1, \ldots, m\}$
- If horse i wins, gambler get payoff o_i per 1
- Gambler strategy $\mathbf{b} = (b_1, \ldots, b_m)$ — b_i is the fraction of gambler wealth invested in horse i ($bi \geq 0$ and $\sum_i b_i = 1$)

Horse Racing

- ▶ Horses $\{1, \ldots, m\}$

- ▶ If horse i wins, gambler get payoff o_i per 1

- ▶ Gambler strategy $\mathbf{b} = (b_1, \ldots, b_m)$ — b_i is the fraction of gambler wealth invested in horse i ($b_i \geq 0$ and $\sum_i b_i = 1$)

- ▶ If horse i wins, gamblers' wealth is multiplied by $b_i o_i$

Horse Racing

- ▶ Horses $\{1, \ldots, m\}$

- ▶ If horse i wins, gambler get payoff o_i per 1

- ▶ Gambler strategy $\mathbf{b} = (b_1, \ldots, b_m)$ — b_i is the fraction of gambler wealth invested in horse i ($b_i \geq 0$ and $\sum_i b_i = 1$)

- ▶ If horse i wins, gamblers' wealth is multiplied by $b_i o_i$

- ▶ Let $X \sim (p_1, \ldots, p_m)$ be the outcome of a random race.

Horse Racing

- Horses $\{1, \ldots, m\}$

- If horse i wins, gambler get payoff o_i per 1

- Gambler strategy $\mathbf{b} = (b_1, \ldots, b_m)$ — b_i is the fraction of gambler wealth invested in horse i ($bi \geq 0$ and $\sum_i b_i = 1$)

- If horse i wins, gamblers' wealth is multiplied by $b_i o_i$

- Let $X \sim (p_1, \ldots, p_m)$ be the outcome of a random race.

- $S(X) := \mathbf{b}(X)\mathbf{o}(X)$ is the factor in which gamblers' wealth is multiplied in a single race (letting $\mathbf{z}(i) = z_i$)

Horse Racing

- Horses $\{1, \ldots, m\}$

- If horse i wins, gambler get payoff o_i per 1

- Gambler strategy $\mathbf{b} = (b_1, \ldots, b_m)$ — b_i is the fraction of gambler wealth invested in horse i ($b_i \geq 0$ and $\sum_i b_i = 1$)

- If horse i wins, gamblers' wealth is multiplied by $b_i o_i$

- Let $X \sim (p_1, \ldots, p_m)$ be the outcome of a random race.

- $S(X) := \mathbf{b}(X)\mathbf{o}(X)$ is the factor in which gamblers' wealth is multiplied in a single race (letting $\mathbf{z}(i) = z_i$)

- We are interested in $S_n := \prod_{i=1}^{n} S(X_i)$, where X_i's are iid $\sim p$

Doubling rate

For gambling strategy **b**, and race outcome **p**,
$S_n := \prod_{i=1}^{n} S(X_i) = \prod_{i=1}^{n} \mathbf{b}(X_i)\mathbf{o}(X_i)$, where X_i's are iid $\sim p$

Doubling rate

For gambling strategy **b**, and race outcome **p**,
$S_n := \prod_{i=1}^n S(X_i) = \prod_{i=1}^n \mathbf{b}(X_i)\mathbf{o}(X_i)$, where X_i's are iid $\sim p$

Definition 7 (doubling rate)

The doubling rate is $W(\mathbf{b}, \mathbf{p}) = \sum_{i=1}^m p_i \log(b_i o_i)$

Doubling rate

For gambling strategy **b**, and race outcome **p**,
$S_n := \prod_{i=1}^n S(X_i) = \prod_{i=1}^n \mathbf{b}(X_i)\mathbf{o}(X_i)$, where X_i's are iid $\sim p$

Definition 7 (doubling rate)

The doubling rate is $W(\mathbf{b}, \mathbf{p}) = \sum_{i=1}^m p_i \log(b_i o_i)$

Theorem 8

For race outcome $\sim \mathbf{p}$ and gambling strategy $\mathbf{b}$, it holds that $S_n \xrightarrow{n} 2^{nW(\mathbf{b},\mathbf{p})}$

Doubling rate

For gambling strategy **b**, and race outcome **p**,
$S_n := \prod_{i=1}^{n} S(X_i) = \prod_{i=1}^{n} \mathbf{b}(X_i)\mathbf{o}(X_i)$, where X_i's are iid $\sim p$

Definition 7 (doubling rate)

The doubling rate is $W(\mathbf{b}, \mathbf{p}) = \sum_{i=1}^{m} p_i \log(b_i o_i)$

Theorem 8

*For race outcome $\sim \boldsymbol{p}$ and gambling strategy **b**, it holds that $S_n \xrightarrow{n} 2^{nW(\boldsymbol{b},\boldsymbol{p})}$*

Proof:

Doubling rate

For gambling strategy **b**, and race outcome **p**,
$S_n := \prod_{i=1}^{n} S(X_i) = \prod_{i=1}^{n} \mathbf{b}(X_i)\mathbf{o}(X_i)$, where X_i's are iid $\sim p$

Definition 7 (doubling rate)

The doubling rate is $W(\mathbf{b}, \mathbf{p}) = \sum_{i=1}^{m} p_i \log(b_i o_i)$

Theorem 8

For race outcome $\sim \mathbf{p}$ and gambling strategy $\mathbf{b}$, it holds that $S_n \xrightarrow{n} 2^{nW(\mathbf{b},\mathbf{p})}$

Proof:

- ▸ fix **p** and **b** and let $X_1, \ldots, X_m$ be iid $\sim \mathbf{p}$

Doubling rate

For gambling strategy **b**, and race outcome **p**,
$S_n := \prod_{i=1}^n S(X_i) = \prod_{i=1}^n \mathbf{b}(X_i)\mathbf{o}(X_i)$, where X_i's are iid $\sim p$

Definition 7 (doubling rate)

The doubling rate is $W(\mathbf{b}, \mathbf{p}) = \sum_{i=1}^m p_i \log(b_i o_i)$

Theorem 8

For race outcome $\sim \mathbf{p}$ and gambling strategy $\mathbf{b}$, it holds that $S_n \xrightarrow{n} 2^{nW(\mathbf{b},\mathbf{p})}$

Proof:

- fix **p** and **b** and let $X_1, \ldots, X_m$ be iid $\sim \mathbf{p}$
- $\log S(X_1), \ldots, \log S(X_n)$ are iid

Doubling rate

For gambling strategy **b**, and race outcome **p**,
$S_n := \prod_{i=1}^{n} S(X_i) = \prod_{i=1}^{n} \mathbf{b}(X_i)\mathbf{o}(X_i)$, where X_i's are iid $\sim p$

Definition 7 (doubling rate)

The doubling rate is $W(\mathbf{b}, \mathbf{p}) = \sum_{i=1}^{m} p_i \log(b_i o_i)$

Theorem 8

For race outcome $\sim \mathbf{p}$ and gambling strategy $\mathbf{b}$, it holds that $S_n \xrightarrow{n} 2^{nW(\mathbf{b},\mathbf{p})}$

Proof:

- ▶ fix **p** and **b** and let $X_1, \ldots, X_m$ be iid $\sim \mathbf{p}$
- ▶ $\log S(X_1), \ldots, \log S(X_n)$ are iid
- ▶ By weak low of large numbers,

$$\frac{1}{n} \log S_n = \frac{1}{n} \sum_i \log(S(X_i)) \xrightarrow{n} \mathbb{E}(\log S(X_1)) = W(\mathbf{b}, \mathbf{p})$$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{p}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{b}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{p}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

$$W(\mathbf{b}, \mathbf{p}) = \sum_{i=1^m} p_i \log(b_i o_i)$$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{b}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

$$
\begin{aligned}
W(\mathbf{b}, \mathbf{p}) &= \sum_{i=1^m} p_i \log(b_i o_i) \\
&= \sum_i p_i \log\left(\frac{b_i}{p_i} p_i o_i\right)
\end{aligned}
$$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{p}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

$$
\begin{aligned}
W(\mathbf{b}, \mathbf{p}) &= \sum_{i=1^m} p_i \log(b_i o_i) \\
&= \sum_i p_i \log\left(\frac{b_i}{p_i} p_i o_i\right) \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - \sum_i p_i \cdot \log \frac{b_i}{p_i}
\end{aligned}
$$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{p}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

$$
\begin{aligned}
W(\mathbf{b}, \mathbf{p}) &= \sum_{i=1^m} p_i \log(b_i o_i) \\
&= \sum_i p_i \log\left(\frac{b_i}{p_i} p_i o_i\right) \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - \sum_i p_i \cdot \log \frac{b_i}{p_i} \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - D(\mathbf{p} \| \mathbf{b})
\end{aligned}
$$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{p}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

$$
\begin{aligned}
W(\mathbf{b}, \mathbf{p}) &= \sum_{i=1^m} p_i \log(b_i o_i) \\
&= \sum_i p_i \log \left(\frac{b_i}{p_i} p_i o_i \right) \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - \sum_i p_i \cdot \log \frac{b_i}{p_i} \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - D(\mathbf{p}||\mathbf{b}) \\
&\leq \sum_i p_i \log o_i - H(\mathbf{b})
\end{aligned}
$$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{p}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

$$
\begin{aligned}
W(\mathbf{b}, \mathbf{p}) &= \sum_{i=1^m} p_i \log(b_i o_i) \\
&= \sum_i p_i \log \left(\frac{b_i}{p_i} p_i o_i \right) \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - \sum_i p_i \cdot \log \frac{b_i}{p_i} \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - D(\mathbf{p}||\mathbf{b}) \\
&\leq \sum_i p_i \log o_i - H(\mathbf{b}) = W(\mathbf{p}, \mathbf{p})
\end{aligned}
$$

Maximal doubling rate

Theorem 9

Let $W^*(\boldsymbol{p}) = \max_{\boldsymbol{p}} W(\boldsymbol{b}, \boldsymbol{p})$, then $W^*(\boldsymbol{p}) = W(\boldsymbol{p}, \boldsymbol{p}) = \sum_i p_i \log o_i - H(\boldsymbol{p})$

Roughly, best strategy is to follow the distribution (ignoring the payoffs)!

$$
\begin{aligned}
W(\mathbf{b}, \mathbf{p}) &= \sum_{i=1^m} p_i \log(b_i o_i) \\
&= \sum_i p_i \log \left(\frac{b_i}{p_i} p_i o_i \right) \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - \sum_i p_i \cdot \log \frac{b_i}{p_i} \\
&= \sum_i p_i \log o_i - H(\mathbf{p}) - D(\mathbf{p}||\mathbf{b}) \\
&\leq \sum_i p_i \log o_i - H(\mathbf{b}) = W(\mathbf{p}, \mathbf{p})
\end{aligned}
$$

where $D(\mathbf{p}||\mathbf{b})$, the relative entropy from $\mathbf{p}$ to $\mathbf{b}$, is known to be non-negative.

Gambling with side information

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y)o(x) \right)$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

Gambling with side information

▶ Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

▶ $W^*(X) := \max_\mathbf{b} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

The best strategy for $(X, \mathbf{o})$

▶ $W^*(X|Y) := \max_\mathbf{b} \sum_{x,y} p(x, y) \log(b(x|y) o(x))$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \, (p(x|y)o(x))$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x, y) \log(b(x|y)o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x, y) \log(b(x|y) o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x, y) \log(b(x|y) o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x, y) \log(b(x|y) o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Theorem 10

$\Delta W = I(X; Y)$.

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y)o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x, y) \log(b(x|y)o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Theorem 10

$\Delta W = I(X; Y)$.

- $W^*(X) = \sum_x p_X(x) \log o(x) - H(X)$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x, y) \log(b(x|y) o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Theorem 10

$\Delta W = I(X; Y)$.

- $W^*(X) = \sum_x p_X(x) \log o(x) - H(X)$
- $W^*(X|Y) = \sum_{x,y} p(x, y) \log \left(p(x|y) o(x) \right)$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x, y) \log(b(x|y) o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Theorem 10

$\Delta W = I(X; Y)$.

- $W^*(X) = \sum_x p_X(x) \log o(x) - H(X)$
- $W^*(X|Y) = \sum_{x,y} p(x, y) \log \left(p(x|y) o(x) \right)$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_\mathbf{b} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_\mathbf{b} \sum_{x,y} p(x, y) \log(b(x|y) o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Theorem 10

$\Delta W = I(X; Y)$.

- $W^*(X) = \sum_x p_X(x) \log o(x) - H(X)$
- $W^*(X|Y) = \sum_{x,y} p(x, y) \log \left(p(x|y) o(x) \right) = \sum p_X(x) \log o(x) - H(X|Y)$

Gambling with side information

- Let $(X, Y) \sim p$ be the outcome of a race and a side information, and let **o** be the race payoffs.

- $W^*(X) := \max_{\mathbf{b}} \sum_x p_X(x) \left(p(x|y) o(x) \right)$

  The best strategy for $(X, \mathbf{o})$

- $W^*(X|Y) := \max_{\mathbf{b}} \sum_{x,y} p(x,y) \log(b(x|y) o(x))$

  The best strategy for $(X, \mathbf{o})$, when Y is known

- $\Delta W := W^*(X|Y) - W^*(X)$

Theorem 10

$\Delta W = I(X; Y)$.

- $W^*(X) = \sum_x p_X(x) \log o(x) - H(X)$

- $W^*(X|Y) = \sum_{x,y} p(x,y) \log \left(p(x|y) o(x) \right) = \sum p_X(x) \log o(x) - H(X|Y)$

- Hence, $\Delta W = H(X) - H(X|Y) = I(X; Y)$. $\qquad \square$